CPE403 – Advanced Embedded Systems

Design Assignment 1

Name: Jenifer Christina

Email: chrisj14@unlv.nevada.edu

Github Repository link: https://github.com/chrisj14/CCS-Assignment

Youtube Playlist link: https://youtu.be/-lvkrEkpGHY

1. Code for Tasks. for each task submit the modified or included code (from the base code) with highlights and justifications of the modifications. Also include the comments. If no base code is provided, submit the base code for the first task only. Use separate page for each task.

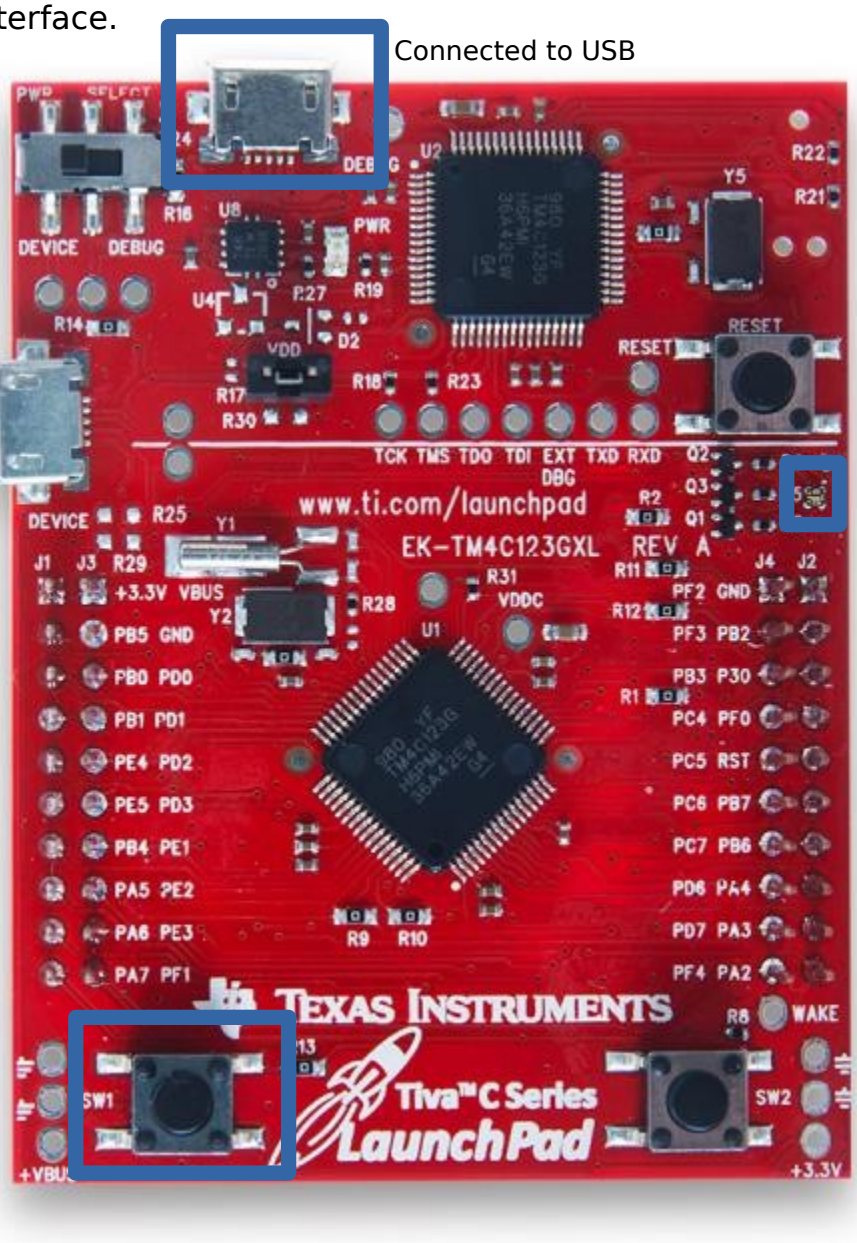
Task 01:
a) Continuously display the temperature of the device (internal temperature sensor) on the terminalusing a timer interrupt every 0.5 secs,
b) Using PF4button interrupt toggle all (RGB) LEDs.(PS: your program will have two interrupts)

```
/* ----------------------          Include Files       ---------------------- */
#include <stdint.h>             // Library of Standard Integer Types
#include <stdbool.h>            // Library of Standard Boolean Types
#include "inc/tm4c123gh6pm.h"   //def. for the interrupt and register assignments on
the Tiva C Series device on the launchPad board
#include "inc/hw_memmap.h"      // Macros defining the memory map of the Tiva C Series
device
#include "inc/hw_gpio.h"        // Defines macros for GPIO hardware
#include "inc/hw_types.h"       // Defines common types and macros
#include "inc/hw_ints.h"
#include "driverlib/sysctl.h"   // Defines and macros for System Control API of
DriverLib
#include "driverlib/pin_map.h"  // Mapping of peripherals to pins of all parts
#include "driverlib/gpio.h"     // Defines and macros for GPIO API of DriverLib
#include "driverlib/interrupt.h"//defines & macros for NVIC Controller(Interrupt)API
of driverlib.
#include "driverlib/timer.h"    //Defines and macros for Timer API of driverLib.
#include "driverlib/sysctl.h"
#include "driverlib/adc.h"
#include "driverlib/debug.h"
#include "driverlib/uart.h"
#include "utils/uartstdio.h"
#include <string.h>

#ifdef DEBUG
```

```
void_error_(vhar *pcFilename, uint32_t ui32Line){}
#endif

/* ----------------------      Global Variables        ---------------------- */
uint32_t ui32Period;
char buffer [4];
uint32_t ui32ADC0Value[4];
volatile uint32_t ui32TempAvg;
volatile uint32_t ui32TempValueC;
volatile uint32_t ui32TempValueF;

/* ----------------------     Function Prototypes      ---------------------- */
void GPIOF0IntHandler(void);
void TimerIntHandler(void);

/* ----------------------          Main Program        ---------------------- */
int main(void){

    //System clock to 40Mhz (PLL= 400Mhz / 10 = 40Mhz)
    SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_XTAL_16MHZ|SYSCTL_OSC_MAIN);

    //Configure peripherals
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
    SysCtlPeripheralEnable(SYSCTL_PERIPH_UART0);
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOA);
    SysCtlPeripheralEnable(SYSCTL_PERIPH_TIMER1);
    SysCtlPeripheralEnable(SYSCTL_PERIPH_ADC0);

    //Setup for ADC
    ADCHardwareOversampleConfigure(ADC0_BASE, 32);
    ADCSequenceConfigure(ADC0_BASE, 2, ADC_TRIGGER_PROCESSOR, 0);
    ADCSequenceStepConfigure(ADC0_BASE, 2, 0, ADC_CTL_TS);
    ADCSequenceStepConfigure(ADC0_BASE, 2, 1, ADC_CTL_TS);
    ADCSequenceStepConfigure(ADC0_BASE, 2, 2, ADC_CTL_TS);
    ADCSequenceStepConfigure(ADC0_BASE, 2, 3, ADC_CTL_TS|ADC_CTL_IE|ADC_CTL_END);
    ADCSequenceEnable(ADC0_BASE, 2);
    //Setup for TIMER1
    TimerConfigure(TIMER1_BASE, TIMER_CFG_PERIODIC);
    ui32Period = SysCtlClockGet()/2;     //Period of .5s 2Hz
    TimerLoadSet(TIMER1_BASE, TIMER_A, ui32Period -1);
    IntEnable(INT_TIMER1A);
    TimerIntEnable(TIMER1_BASE, TIMER_TIMA_TIMEOUT);
    //Setup for UART
    GPIOPinConfigure(GPIO_PA0_U0RX);
    GPIOPinConfigure(GPIO_PA1_U0TX);
    GPIOPinTypeUART(GPIO_PORTA_BASE, GPIO_PIN_0 | GPIO_PIN_1);
    UARTClockSourceSet(UART0_BASE, UART_CLOCK_PIOSC);
    UARTStdioConfig(0, 115200, 16000000);
    //Enable Interrupts
    TimerEnable(TIMER1_BASE, TIMER_A);
    ADCSequenceEnable(ADC0_BASE,2);

    //Setup for SW2 Toggle
    //set LEDS connected to pins as outputs
    GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3);
    //Unlock Pin F4 to use an interrupt on SW1
    SYSCTL_RCGC2_R |= 0x00000020;     // activate clock for Port F
    GPIO_PORTF_LOCK_R = 0x4C4F434B;   // unlock GPIO Port F
    GPIO_PORTF_CR_R = 0x1F;           // allow changes to PF4-0
    // only PF0 needs to be unlocked, other bits can't be locked
    GPIO_PORTF_AMSEL_R = 0x00;        // disable analog on PF
```

```
    GPIO_PORTF_PCTL_R = 0x00000000;   // PCTL GPIO on PF4-0
    GPIO_PORTF_DIR_R = 0x0E;          // PF4,PF0 in, PF3-1 out
    GPIO_PORTF_AFSEL_R = 0x00;        // disable alt funct on PF7-0
    GPIO_PORTF_PUR_R = 0x11;          // enable pull-up on PF0 and PF4
    GPIO_PORTF_DEN_R = 0x1F;          // enable digital I/O on PF4-0
    //register the interrupt handler for PF4
    GPIOIntRegister(GPIO_PORTF_BASE, GPIOF0IntHandler);
     //SW1 goes low when pressed
    GPIOIntTypeSet(GPIO_PORTF_BASE, GPIO_PIN_4, GPIO_FALLING_EDGE);
    //enable interrupts on PF4
    GPIOIntEnable(GPIO_PORTF_BASE, GPIO_PIN_4);

    //Enable master interrupt
    IntMasterEnable();


    while(1)
    {
    }
}

void GPIOF0IntHandler(void) //interrupt handler for GPIO pin F4
{
    //clear interrupt flag on pin F4
    GPIOIntClear(GPIO_PORTF_BASE, GPIO_PIN_4);
    //Toggle all RGB LEDs
    if (GPIOPinRead(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3))
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 0);
    else
        GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, GPIO_PIN_1|
GPIO_PIN_2|GPIO_PIN_3);
}

void TimerIntHandler(void)
{
    // Clear Timer Interrupt
    TimerIntClear(TIMER1_BASE, TIMER_TIMA_TIMEOUT);

    ADCIntClear(ADC0_BASE,2);
    ADCProcessorTrigger(ADC0_BASE, 2);

    ADCSequenceDataGet(ADC0_BASE, 2, ui32ADC0Value);

    ui32TempAvg = (ui32ADC0Value[0] + ui32ADC0Value[1] + ui32ADC0Value[2] +
ui32ADC0Value[3] + 2)/4;
    ui32TempValueC = (1475 - ((2475 * ui32TempAvg)) / 4096)/10;
    ui32TempValueF = ((ui32TempValueC * 9) + 160) / 5;
    UARTprintf("Temperature: ");
    UARTprintf("C %3d\t",ui32TempValueC );
    UARTprintf("F %3d\t",ui32TempValueF );
    UARTprintf("\n");
}
```

2. Block diagram and/or Schematics showing the components, pins used, and interface.



3. Screenshots of the IDE, physical setup, debugging process - Provide screenshot of successful compilation, screenshots of registers, variables, graphs, etc.

Terminal

/dev/ttyACM0

```
Temperature: C  22      F  71
Temperature: C  22      F  71
Temperature: C  22      F  71
Temperature: C  22      F  71
Temperature: C  22      F  71
Temperature: C  23      F  73
Temperature: C  23      F  73
Temperature: C  23      F  73
Temperature: C  23      F  73
Temperature: C  23      F  73
Temperature: C  23      F  73
Temperature: C  23      F  73
Temperature: C  23      F  73
Temperature: C  23      F  73
Temperature: C  23      F  73
Temperature: C  23      F  73
Temperature: C  23      F  73
Temperature: C  23      F  73
Temperature: C  23      F  73
Temperature: C  23      F  73
Temperature: C  23      F  73
```

4. Declaration

I understand the Student Academic Misconduct Policy - http://studentconduct.unlv.edu/misconduct/policy.html

“This assignment submission is my own, original work”.
Jenifer Christina